

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-271764

(43)Date of publication of application : 20.10.1995

(51)Int.Cl.

G06F 17/16

(21)Application number : 06-316800

(71)Applicant : INTERNATL BUSINESS MACH CORP
<IBM>

(22)Date of filing : 20.12.1994

(72)Inventor : ARGARWAL RAMESH CHANDRA
GROVES RANDALL DEAN
GUSTAVSON FRED GEHRUNG
JOHNSON MARK ALAN
OLSSON BRETT

(30)Priority

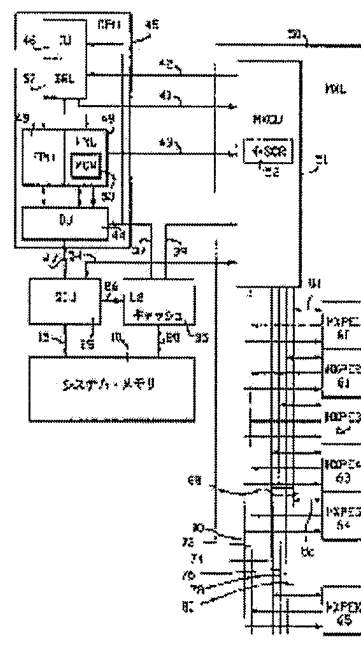
Priority number : 94 217533 ????Priority date : 24.03.1994 ????Priority country : US

(54) COMPUTER PROCESSOR AND SYSTEM

(57)Abstract:

PURPOSE: To provide a computer system provided with parallel processors for simultaneously executing an operation related to the group of data word pairs so as to substantially increase the throughput.

CONSTITUTION: A matrix processor for making high-speed numerical value calculation possible is presented. The processor is a vector processor formed from plural processing elements. A first processor is provided with a set of N pieces of registers and a first element or word among the N pieces of data vectors is stored in the registers. The respective processing elements are provided with an arithmetic unit capable of executing an arithmetic operation relating to the N pieces of elements in a set of N pieces of the registers. The respective data vectors are provided with the K pieces of the elements. Thus, the K pieces of the processing elements are present. For the vector operation of the matrix processor, the same operation is simultaneously executed relating to all the two or more vectors. Within one machine cycle after a preceding vector operation, a succeeding vector operation can be executed.



LEGAL STATUS

[Date of request for examination] 20.12.1994

[Date of sending the examiner's decision of rejection] 02.06.1998

[Kind of final disposal of application other than the
examiner's decision of rejection or application
converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of 10-013597]

rejection]

[Date of requesting appeal against examiner s
decision of rejection]

28.08.1998

[Date of extinction of right]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-271764

(43) 公開日 平成7年(1995)10月20日

(51) Int.Cl.⁶

識別記号

庁内整理番号

F I

技術表示箇所

G 0 6 F 17/16

G 0 6 F 15/ 347

S

審査請求 有 請求項の数14 O L (全 17 頁)

(21) 出願番号 特願平6-316800

(22) 出願日 平成6年(1994)12月20日

(31) 優先権主張番号 2 1 7 5 3 3

(32) 優先日 1994年3月24日

(33) 優先権主張国 米国 (US)

(71) 出願人 390009531

インターナショナル・ビジネス・マシー
ズ・コーポレーション

INTERNATIONAL BUSIN
ESS MACHINES CORPO
RATION

アメリカ合衆国10504、ニューヨーク州
アーモンク (番地なし)

(72) 発明者 ラメシュ・チャンドラ・アーガーワル
アメリカ合衆国ニューヨーク州、ヨークタ
ウン・ハイツ、ガース・コート 650

(74) 代理人 弁理士 合田 潔 (外2名)

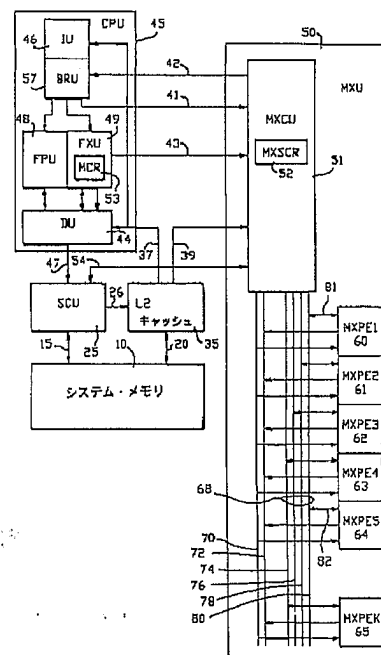
最終頁に続く

(54) 【発明の名称】 計算機プロセッサ及びシステム

(57) 【要約】

【目的】 本発明は改良された計算機システムを提供する。

【構成】 高速度数値計算を可能にするマトリクス処理装置が開示される。その処理装置は複数個の処理エレメントから形成されたベクトル処理装置である。I 番目の処理装置はN個のレジスタのセットを有し、それらレジスタには、N個のデータ・ベクトルのうちのI 番目のエレメント又はワードが記憶される。各処理エレメントは、N個のレジスタのセットにおけるN個のエレメントに関する演算オペレーションを遂行できる演算装置を有する。各データ・ベクトルはK個のエレメントを有する。従って、K個の処理エレメントが存在する。マトリクス処理装置のベクトル・オペレーションは2つ以上のベクトルすべてに関して同じオペレーションを同時に遂行する。先行のベクトル・オペレーションの後の1機械サイクル内で後続のベクトル・オペレーションが遂行可能である。



【特許請求の範囲】

【請求項1】 プロセッサ・サイクル・タイムを有する計算機プロセッサにして、

K個の処理エレメント（但し、 $K > 1$ ）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、 $N > 1$ ）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、 $1 \leq I \leq N$ ）、それによって、N個のベクトルを提供するものと、

前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、

を含む計算機プロセッサ。

【請求項2】 前記N個のレジスタのセットにおける各レジスタはM個のビット記憶ロケーション（但し、 $M > 0$ ）を有することを特徴とする請求項1に記載の計算機プロセッサ。

【請求項3】 前記ベクトル・オペレーションを遂行するための手段である第1演算装置及び前記他のベクトル・オペレーションを遂行するための手段である第2演算装置は同じ演算装置であることを特徴とする請求項1に記載の計算機プロセッサ。

【請求項4】 マトリクス制御装置を含むことを特徴とする請求項1に記載の計算機プロセッサ。

【請求項5】 前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスを含むことを特徴とする請求項4に記載の計算機プロセッサ。

【請求項6】 前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスを含むことを特徴とする請求項4又は5に記載の計算機プロセッサ。

【請求項7】 前記少なくとも1つのデータ・バスは前記K個の処理エレメントのうちの2個以上の処理エレメントによって共用されることを特徴とする請求項5に記載の計算機プロセッサ。

【請求項8】 サイクル・タイムを有する計算機プロセッサにして、

マトリクス制御装置と、
前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段と、

を含む計算機プロセッサ。

【請求項9】 前記手段はベクトル演算オペレーションを遂行することを特徴とする請求項8に記載の計算機プロセッサ。

【請求項10】 前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスと、
前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスと、
を含むことを特徴とする請求項8に記載の計算機プロセッサ。

10 【請求項11】 サイクル・タイムを有する計算機システムにして、

メイン・メモリと、

キャッシュ・メモリと、

記憶制御装置と、

中央処理装置と、

マトリクス処理装置と、

前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、

20 前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、

前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、

前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、

前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、

前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、

30 前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、

前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、

K個の処理エレメント（但し、 $K > 1$ ）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、 $N > 1$ ）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、 $1 \leq I \leq N$ ）、それによって、N個のベクトルを提供するものと、

40 前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、

50 前記所定のサイクル・タイム後の次のサイクルタイムに

において他のベクトル・オペレーションを遂行するための手段と、
を含む計算機システム。

【請求項12】 サイクル・タイムを有する計算機システムにして、
メイン・メモリと、
キャッシュ・メモリと、
記憶制御装置と、
中央処理装置と、
マトリクス処理装置と、

前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、

前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、
を含み、

前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、

前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、

前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、

前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、

前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、

前記マトリクス処理装置は、

K個の処理エレメント（但し、 $K > 1$ ）にして、前記K個の処理エレメントの各々はN個のレジスタのセット

（但し、 $N > 1$ ）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、 $1 \leq I \leq N$ ）、それによって、N個のベクトルを提供するものと、

前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、

前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、

を含むことを、

特徴とする計算機システム。

【請求項13】 マトリクス制御装置と、

前記マトリクス制御装置を前記K個の処理エレメントに接続するためのコマンド・バスと、

前記マトリクス制御装置を前記K個の処理エレメントに

接続するための少なくとも1つのデータ・バスと、
を含むことを特徴とする請求項12に記載の計算機システム。

【請求項14】 サイクル・タイムを有する計算機システムにして、

メイン・メモリと、

中央処理装置と、

マトリクス処理装置と、

前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、

前記メイン・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、

前記中央処理装置は第1データ・バスによって前記メイン・メモリに接続する手段と、

前記マトリクス制御装置は第2データ・バスによって前記メイン・メモリに接続する手段と、

前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、

前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、

K個の処理エレメント（但し、 $K > 1$ ）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、 $N > 1$ ）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、 $1 \leq I \leq N$ ）、それによって、N個のベクトルを提供するものと、

前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、

前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、

を含む計算機システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、概して云えば、データ処理システムの分野に関するものであり、詳しく云えば、本発明は、複数個のプロセッサにおいて数値集中計算(numerically intensive computing)するためのデータ処理システムに関するものである。更に詳しく云えば、本発明は、N個のプロセッサがN個のエレメント・ベクトル上で縮小命令セットの命令を同時に実行する多重プロセッサ・システムに関するものである。

【0002】

【従来の技術】コンピュータにおける計算力は成長してきたし、急速に成長し続けている。この増大した計算力は、新たな方法でコンピュータを使用するための新たな機会をその計算力のユーザに与えており、従って、コンピュータ・パワーを増大させるための更なる要求を生み出している。増大した計算力が特に有用である1つの領域は、数値集中計算の領域である。数値集中計算は、大量のデータに関する限られたオペレーション・セットの計算に関連している。数値集中計算の技法は、イメージ処理、信号処理、人工知能、或いは気象力学又は流体力学のコンピュータ・シミュレーションのような幅広い分野の適用業務において使用される。これらの適用業務における数値集中計算の技法が直面する共通の問題は、各適用業務に関連した大量のデータに関して必要な有限の数の関数を、可能な限りの最小の時間で計算することである。

【0003】初期の従来技術のコンピュータ・システムは、中央処理装置(CPU)、メモリ、及び入出力(I/O)コントローラより成るものであった。CPUは、データ又は命令をメモリに転送するように或いはそれらをメモリから取り出すようにI/Oコントローラに命令するプログラム、及び種々の命令セットに従ってデータを処理するプログラムを実行してした。データを処理することは、CPUが遂行すべき特殊な命令をデコードするようなオペレーション、その特殊な命令を実行するようなオペレーション、及びその結果をメモリに戻すようなオペレーションを含んでいた。CPUは、それがプログラムにおける命令のリストからの命令を一時に1つずつ処理するという点では複雑ではなかった。しかし、このタイプのCPUは、それが一時に1つの命令しか処理しなかったため遅かった。その遅い処理時間に応答して、従来技術のシステムは、プロセッサをパイプライン化することを導入した。

【0004】パイプライン化したプロセッサでは、異なる命令に対して、CPUの種々の独立した機能が一時に生じる。例えば、1つの命令に対するデータを取り出すためのメモリに対するプロセッサの命令は、プロセッサが他の命令のオペレーション・コードをデコードしている時に生じようとしているかもしれない。パイプライン化することは、何れの個々の命令の実行もスピード・アップするものではないが、それは、前の命令が処理を完了してしまう前に後続の命令がCPUにおける処理を開始するため、一連の命令の処理をスピード・アップしている。パイプライン化に加えて、従来技術は、限定された命令セットが数値的問題に対して反復的に使用されることも認識してした。その結果、従来技術は、それら限定された命令セットを素早く実行するために、CPU内に特殊プロセッサの機能的ブロックを導入した。例えば、いくつかのCPUは、加算、乗算、又はブランチだ

けを遂行するための機能的ブロックを含んでいた。これら機能的ブロックは、メインCPUよりもずっと速くこれらの限定された機能を実行したし、これは数値的関数を処理する速度も増加させた。

【0005】パイプライン化又は限定された命令のための別個の機能ユニット、或いはそれらの両方を組み込んだ従来技術のプロセッサでは、単一の命令ストリームが単一のデータ・ストリーム上で動作していた。即ち、各命令は、定義されたデータに関して一時に1つの計算を行うように動作してした。そのようなプロセッサは、単一命令・単一データ処理の略称であるSISDと呼ばれる。SISDタイプのプロセッサの問題は、プログラムのセグメントが多数の多様なエレメントに適用されることをそのプログラムが必要とする場合、そのプログラムは、順次に何回もそのセグメントを循環しなければならぬことである。これは、多くのタイプの数値的問題に対する状況であった。そして、このようなタイプの問題に関するプロセッサ速度の増加を援助するために、SIMD(単一命令・多重データ処理)タイプのプロセッサが開発された。

【0006】SIMDプロセッサ・ユニットは、同じ機能的能力を持った複数個の処理エレメントより成り、そしてそれら処理エレメントは、それぞれ、各処理エレメントが別個のデータ・ストリームに関して動作する時、一時に1つの命令に基づいて動作する。代表的なSIMDプロセッサ・ユニットは、3つの主要な素子、即ち、処理エレメントのアレイ、経路指定ネットワーク、及びコントローラを含む。対照的に、本発明によるプロセッサは経路指定ネットワークを使用せず、その結果、低コストのプロセッサが得られ、メイン・メモリ及び処理エレメントに及びそれらからデータを移動させるための時間の減少が得られる。経路指定ネットワークは、任意の1つの処理エレメントからの結果を、それを必要とするそのアレイ内の他の任意の処理エレメントにコミュニケーションする。コントローラは、そのSIMDプロセッサ・ユニットが一部分となるメインCPUからの命令及びデータに回答して、その処理エレメントのアレイ及び経路指定ネットワークのオペレーションを制御する。

【0007】マルチプロセッサSIMDタイプのシステムによる問題は、コンピュータ・メモリ・システムからデータを十分に速い速度で得るためのマルチプロセッサ・アレイの能力と多数のプロセッサの使用を最適化するためのメインCPUの能力とによって、システム・パフォーマンスが制限されるということである。とりわけ、メイン・メモリからのデータをアレイにおける各プロセッサに供給するメモリ・システムを構築することは、非常に高いメモリ帯域幅(BW)を必要とし、従って、余りにも高価であり、或いは殆どのアプリケーションにとって非実用的である。更に、個々の各プロセッサがそのアレイにおける他のプロセッサと直接にコミュニケーション

7

する経路指定ネットワークは、そのプロセッサのアレイの全体的なパフォーマンスを下げる。その従来技術は、限られた成功でもってこれらの問題を解決しようとした。更に詳しく言えば、その従来技術は、プロセッサ・アレイにおけるそれら複数個のプロセッサをマトリクス状に接続し、それらプロセッサがそれらのいくつかの最も近接したプロセッサとだけコミュニケーションするようにした。

【0008】この配列は、最も近接したプロセッサがコミュニケーションする能力を増大させ、従って、命令がそれら最も近接したプロセッサ相互間だけのコミュニケーションを必要とする時にはそれらプロセッサの全体的なパフォーマンスを増大させる。しかし、殆どの命令は無作為なプロセッサ相互間のコミュニケーションを必要とするものであり、最も近接したプロセッサを介した無作為なプロセッサへの経路指定コミュニケーションは、そのプロセッサ・アレイの全体的なパフォーマンスを減少させる。その結果、プロセッサの近接プロセッサ接続は、実行される命令の数及びタイプによっては、実際に、マルチプロセッサ・アレイのパフォーマンスを減少させる。各プロセッサが多数の他のプロセッサに直接に接続しているハイパーキューブ・タイプの配列に複数のプロセッサが接続させる時、それらプロセッサ相互間のコミュニケーションの速度は増加する。このタイプの配列によって、直接に相互接続されてないプロセッサ相互間のコミュニケーションを必要とすることが少なくなり、そのプロセッサ・アレイのパフォーマンスが増加するであろう。しかし、プロセッサ・アレイ・パフォーマンスの増加は、依然として、そのプロセッサ・アレイに関連したメモリ帯域幅によって制限される。プロセッサ・アレイは、データを個々のプロセッサに配送し得る速さで命令を実行することはできる。ハイパーキューブ配列は、そのようなメモリ・システムをそのハイパーキューブ配列と結合して実施することが多くのコンピュータ・システムにおいて非実用的であるような高いメモリ帯域幅を必要とする。

【0009】最近の高パフォーマンスのコンピュータにとって、パフォーマンスは主としてその機械のサイクル・タイムによって制限される。全体的なパフォーマンスを増加させるために、しばしば、並列化が使用される。多数のプロセッサと共に共用メモリの並列システムを構築することが非常に困難であることはわかっている。最近、いくつもの分散型メモリ並列システムが構築されている。Intel Paragon, Cray T3-D, IBM SP1はこのクラスの機械の例である。一般には、多数のプロセッサでもって分散型メモリ・システムを最適に利用すること及びそのソフトウェアを書くことは更に困難である。これらの機械は、しばしば、通信障害によって遅滞させられる。又、殆どのシステム・コンポーネントの複製及び通信ネットワークの付加的

8

コスト及びオーバーヘッドのために、これらの高度な並列機械の価格性能比は、基礎的なユニプロセッサに対する価格性能比ほど有利なものではないことが多い。

【0010】本発明によるコンピュータ・システムにおける2つの主要な考察点は、命令の実行の制御とメモリ管理である。制御の考察点及びオペレーティング・システムの観点から云えば、それは、単一の命令ストリームしか実行されない場合、又はメモリの1つのイメージしか維持されない場合、大変望ましい。換言すれば、オペレーティング・システムの観点から云えば、その機械が単一のCPUの機械のようにみえることが望ましい。本発明は、「単一の命令ストリーム及びメモリ・マップ」モデルの傘の下で同時に動作しながら、或クラスのNIC（数値集中計算）問題に関して非常に高いパフォーマンスを得るシステム構成を定義するためのものである。

【0011】最近の高パフォーマンス・コンピュータにとって、パフォーマンスは、メモリ・システムから得られる帯域幅によっても制限される。更に、本発明は、或クラスの数値集中計算の問題に対するメモリ帯域幅の要件を少なくするために、上記のシステム構造で働くアルゴリズム構造を提供する。

【0012】問題の数値的強度を数値化する1つの簡単な測定法は、浮動小数点演算（フロップス）の合計数をカウントすること、及びその計算に関連したデータ・ポイントの合計数（又は、必要なメモリ・スペースの量）によってそれを除することである。この比率を「浮動小数点演算対データ比」と呼ぶことにする。要するに、この比をFDRと表すことにする。FDRはその問題に対して全体として計算可能であり、その問題によって包含される副次問題に対しても計算可能である。一般的には、副次問題に対するFDRは、その問題全体に対するFDRよりもかなり小さい。初歩的なスカラ演算レベルでは、FDRは1よりも小さいであろう。殆どのNIC問題に対して、当初の問題を適当にブロック化（アルゴリズム的に）して、種々なレベルのメモリ階層に適合した副次問題にすることによって、FDRはかなり改良可能である。

【0013】一般的には、より大きなメモリ・スペースが所与のレベルにおいて利用可能にされる場合、より高い値のFDRが達成可能である。階層メモリ・システムでは、所与のレベルのメモリにおけるメモリ帯域幅（BW）の要件は、そのステージにおけるFDR比に逆比例する。最高のレベル（最大のメモリ・サイズ）では、帯域幅要件は最も小さく、そしてより低い（及びより小さい）メモリのレベルに進むにつれて、帯域幅要件は次第に増大する。その正確な数は、その問題、使用されるアルゴリズム、及び利用可能な計算資源の量に依存するであろう。一般的には、計算資源が高くなるほど、帯域幅要件も高くなると言える。以下では、メモリの帯域幅要件を大きく増大させることなく、大量の計算資源が最適

に利用可能であるというアルゴリズム的な制限のない例を示す。一般には、アルゴリズムは種々なレベルのメモリ及びそれらの関連した帯域幅を最適に利用するように適合可能である。

【0014】

【発明が解決しようとする課題】本発明の目的は、改良された計算機システムを提供することにある。

【0015】本発明のもう1つの目的は、数値集中計算の問題に対する改良された計算機システムを提供することにある。

【0016】本発明の更にもう1つの問題は、多数のプロセッサを持った改良された計算機システムを提供することにある。

【0017】本発明の更にもう1つの目的は、制限された数の命令をより速く実行するための改良された計算機システムを提供することにある。

【0018】本発明の更にもう1つの目的は、計算機システムのメモリ帯域幅を大きく増大させることなく、制限された数の命令をより速く実行するための改良された計算機システムを提供することにある。

【0019】本発明の更にもう1つの目的は、制限された数の命令をより速く実行するための改良された廉価の計算機システムを提供することにある。

【0020】

【課題を解決するための手段】本発明の幅広く見た場合の視点は、計算機システムのスループットを大幅に増大させるためにデータ・ワード対のグループに関するオペレーションを同時に遂行する並列プロセッサを持った計算機システムである。

【0021】本発明の幅広く見た場合のもう1つの視点は、先行のベクトル・オペレーションの1サイクル・タイム内で後続のベクトル・オペレーションを遂行する計算機システムである。

【0022】本発明の更に特殊な視点は、プロセッサ・サイクル・タイムを持った計算機プロセッサである。計算機プロセッサは、K個の処理エレメント（但し、 $K > 1$ ）を有する。それらK個のエレメントの各々は、N個のレジスタのセット（但し、 $N > 1$ ）及び演算装置を有する。それらレジスタの各々のI番目のエレメントは、ベクトル長KのI番目のベクトル・レジスタに対する記憶ロケーションを与える。その計算機プロセッサは、N個のベクトルのうちの少なくとも1つにおける又はそれらの間のベクトル演算オペレーションを所定のサイクル・タイムで遂行するための手段及びN個のベクトルのうちの1つにおけるオペレーションの結果を記憶するための手段を有する。この場合、ベクトル演算オペレーションは、N個のベクトルのうちの少なくとも1つについて、K個のエレメントすべてに対するK個の演算装置によって実質的に同時に遂行される。その計算機プロセッサは、所定のサイクル・タイム後の次のサイクルタイム

においても1つのベクトル・オペレーションを遂行するための手段を有する。

【0023】

【実施例】メモリ帯域幅（BW）の問題を処理した後、次の問題は、その機械の所与のサイクル・タイムに対して大量の計算を行い得るような計算機そのものの編成である。パフォーマンスの1つの測定法は、1サイクル当たりの浮動小数点演算（FPC）である。IBM S/390のような伝統的なスカラー・アーキテクチャに対しては、1つの浮動小数点演算を発生するためにいくつかのサイクルが必要とされる。先ず、オペランドをレジスタ（又は、浮動小数点装置）にフェッチする必要がある、これにパイプライン化した演算が続き、しかる後、その結果がメイン・メモリに戻されて記憶される。そのようなシステムに対するFPCは、典型的には、わずかな端数である。IBM RS/6000のような機械は多数の機能的ユニットを有し、その各々は調整された態様で1つの指定された機能を行い、その結果、理想的な環境下では1サイクル当たり2又はそれ以上の浮動小数点演算を生じる。ベクトル機械も、長いベクトルにおけるパイプライン待ち時間を隠蔽することによって同様の結果を得る。

【0024】伝統的なベクトル・アーキテクチャでは、同じオペレーションが、独立したオペランドのセットに関して遂行される。独立したオペランドの数はベクトル長と呼ばれる。現在のベクトル機械では、これらオペランドは、パイプライン化した演算装置によって逐次処理される。或機械では、スループット（FPC）を増加させるために、演算パイプと呼ばれる多数の演算装置が設けられる。一般的には、多くのパイプが加えられると、メモリ帯域幅もそれに従って増加する。

【0025】本発明は、オペランドの各セット（そのベクトルの各エレメント）が独立した演算装置によって処理されるシステム構造である。その場合、利用可能な演算装置の数はベクトル長に等しい。S/390の用語では、それはベクトル・セクション・サイズ（VSS又はK）と呼ばれる。ベクトル実行のために必要なサイクルの数はパイプライン長に等しくなるであろう。1つのパイプでは、加算のようなオペレーションは、典型的には、2又はそれ以上のステップに分割される。その場合、各ステップは1機械サイクルで行われる。パイプライン長の遅れは、そのパイプへの入力とそのオペレーションの最終結果を伴うパイプの出力との間の機械サイクルにおける遅れである。しかし、演算装置のパイプラインの性質のために、適当な命令スケジューリングによって、各サイクルで「ベクトル・オペランド」のセットを送りそして「ベクトル結果」得ることが可能でなければならない。これは、SIMD（単一命令・多重データ処理）並列処理の1つの形式である。しかし、それは、米国のThinking Machine社のCM-20

0、CM-5等のような伝統的なSIMD機械とは非常に異なっている。そのようなアーキテクチャにとって、得ることのできる1サイクル当たりの最大の浮動小数点演算は $2 * VSS$ （乗算・加算オペレーションが2浮動小数点演算としてカウントされる）である。更に、そのパフォーマンスのレベルはパラメータ VSS 又は K によってパラメータ化される。理論的には、これは達成可能な最良のパフォーマンスである。個のレベルのパフォーマンスを実際に得ることは、データを送ること及び演算装置からの結果を非常に高い速度で記憶することを必要とする。これを次に説明する。

【0026】伝統的なベクトル・アーキテクチャでは、オペランドを一組のベクトル・レジスタ又はメモリから得ることができる。RISCスカラ・アーキテクチャでは、すべてのオペランドが、先ず、一組のスカラ・レジスタにロードされる。演算装置は、すべてのオペランドをレジスタから取得し、すべての結果をレジスタに戻して記憶する。これは、演算装置に対するメモリへの又はメモリからの2ステップ・データ・パスである。本発明の好適な実施例では、処理エレメントはRISCベクトル・アーキテクチャを使用する。そのアーキテクチャでは、すべての演算装置がベクトル・レジスタとだけ対話し（それらのオペランドをベクトル・レジスタのみから取得し、それらの結果をベクトル・レジスタのみに記憶し）、一方、ベクトル・レジスタはメモリ・システムと対話する。

【0027】RISCアーキテクチャは、1990年1月発行の「IBM Journal of Research and Development」第34巻、第1号の記事に記載されている。これは他のレベルの減結合を直接に提案している。演算装置は、その演算装置と同じインデックス番号を持ったベクトル・レジスタのエレメントだけに対話する必要がある。便宜上、演算装置に0から $K-1$ までの番号を付することにする。 N （もう1つのパラメータ）個のベクトル・レジスタがあるものと仮定する。その場合、演算装置は、 N 個のスカラ・レジスタとだけ対話する必要がある。最良の方法では、これらのレジスタ及び演算装置は同じチップ上にあるものと考えられる。演算装置とレジスタとのこの組合せを処理エレメント（PE）と呼ぶことにする（エレメントの合計数 $=N * K$ ）。 N 個のベクトル・レジスタが K 個のPE上に分配される。この K 個のPEの集合体は、NICA（数値集中計算アクセレータ）と呼ばれる。データがNICAレジスタにある場合、それは1サイクル当たり $2 * K$ 浮動小数点演算のピーク・パフォーマンスを得ることができる。

【0028】図1は、本発明に従ってマトリクス装置（MXU）（或いは、ベクトル・プロセッサ）を使用したコンピュータ・システムの1つの実施例を示す。このコンピュータ・システムでは、中央処理装置（CPU）45及びMXU50が、記憶制御装置（SCU）25及

びキャッシュ・バッファ・メモリ（L2キャッシュ）35を介してメイン・メモリ（メモリ・システム）10に接続される。CPU45は、MXU命令バス43、MXUアドレス及びカウンタ・バス42、及びMXU記憶リクエストIDバス41を介してMXU50に接続される。SCU25は、メモリ・アドレス・バス15によってメイン・メモリ10に接続され、アドレス・バス47によってCPU45に接続される。L2キャッシュ35は、メモリ・データ・バス20によってメイン・メモリ10に接続される。更に、そのL2キャッシュ35は、CPUデータ・バス37によってCPU45に接続され、MXUデータ・バス39によってMXU50に接続される。CPU45は、ブランチ装置（BRU）57、浮動小数点装置（FPU）48、固定小数点装置（FXU）49、レベル1命令キャッシュ装置（IU）46、及びレベル1データ・キャッシュ装置（DU）44より成る汎用プロセッサである。そのIU46及びDU44は任意選択のものであり、単にパフォーマンス上の理由で使用されるだけである。

【0029】BRU57は、CPU45及びMXU50に対するコントロール・フロー（次に実行されるべき命令を決定する）を与える。BRU57は、実行されるべき次の命令のメモリ・ロケーションのアドレスを発生する。その命令がIU46に存在する（それが最近アクセスされた）場合、その命令はIU46からフェッチされる。その命令がIU46に存在しない場合、BRU57はそのロケーションに対するリクエストを発生し、L2キャッシュ・アドレス・バス47を介してSCU25にそのリクエストを送る。SCU25は、L2キャッシュ35の内容のディレクトリを持っている。そのリクエストされたメモリ・ロケーションがL2キャッシュ35に存在する場合、SCU25は、L2キャッシュ35におけるそのロケーションをアクセスするのに必要なコントロールを発生し、L2コントロール・バス26を介してそのL2キャッシュにアクセス・コントロールを送る。L2キャッシュ35はそのアクセスを遂行し、CPUデータ・バス37を介してCPU45にそのロケーションの内容を送る。リクエストされたアドレスがL2キャッシュ35に存在しない場合、SCU25はメモリ・アクセス・リクエストを発生し、メモリ・アドレス・バス15を介してメモリ・システム10にそのリクエストを送る。メモリ・システム10はそのリクエストされたロケーションをアクセスし、メモリ・データ・バス20を介してL2キャッシュ35にそのリクエストされたロケーションの内容を送る。L2キャッシュ35は、CPUデータ・バス37を介してCPU45にそのリクエストされたメモリ・ロケーションの内容を送る。IU46は、そのリクエストされたメモリ・ロケーションの内容を受け取ると、ディスパッチ及び実行のための命令をBRU57に送る。

【0030】BRU57は実行されるべき次の命令を調べ、その命令を適当な実行装置にディスパッチする。ブランチ命令は、BRU57によって実行される。スカラ浮動小数点命令は、FPU48によって実行される。ベクトル命令は、MXU50によって実行される。メモリ・アクセス命令は、FXU48及び関連の実行装置により共同して実行される。その場合、FXU48はロード命令又はストア命令のアドレスを発生し、DU44に(FXU及びFPUメモリ・アクセスのために)又はSCU25に(MXUメモリ・アクセスのために)リクエストを送る。ベクトル命令は、BRU57によって、MXU命令バス41を介してMXU50にディスパッチされる。ベクトル・メモリ・アクセス命令は、MXU50及びFXU49の両方にディスパッチされる。

【0031】上述のように、ベクトル・メモリ・アクセス命令は、FXU49及びMXU50によって共同して実行される。FXU49は、メモリ・アクセスのアドレスを計算し、メモリ・リクエストをDU44に送る。DU44は、リクエストされたロケーションがL1データ・キャッシュに存在しないことを確認するためにキャッシュ・ディレクトリ・ルックアップを行う。そのメモリ・ロケーションが存在する場合、そのロケーションは、先ず、L2キャッシュ35にフラッシュ・アウトされてそれをMXUメモリ・アクセスにとって明白なものにする。そこで、DU44はメモリ・アクセス・リクエストをSCU25に送る。命令メモリ・アクセスと同様に、SCU25は、そのリクエストされたメモリ・ロケーションがL2キャッシュ35に存在するかどうかを決定するためにL2キャッシュ・ディレクトリを調べる。そのロケーションがL2キャッシュ35に存在しない場合、メモリ・システム10におけるそのメモリ・ロケーションがリクエストされ、その内容がメモリ・データ・バス20を介して戻され、L2キャッシュ35に置かれる。

【0032】MXUロード命令に対しては、そのリクエストされたメモリ・ロケーションの内容が、MXUデータ・バス39を介してMXU50に戻される。SCU25は、ロード・データが、MXU-L2キャッシュ・コントロール・バス54を使用してMXUデータ・バス39上に得られることを、MXU50に知らせる。MXUストア命令に対しては、SCU25及びL2キャッシュ35は、そのストア命令のデータがMXU50によってMXUデータ・バス39上に存在するまで待つ。その時点で、そのデータはL2キャッシュ35におけるそのアドレスされたメモリ・ロケーションに置かれる。マトリクス制御装置(MXCU)51は、ストア・データがMXU-L2キャッシュ・コントロール・バス54を使用してMXUデータ・バス39上に得られることをSCU25に知らせる。

【0033】MXU命令バス41を介してディスパッチされるベクトル命令に対して、マトリクス制御装置(M

XCUCU)51は実行されるべき命令を受け取り、そしてその後の実行のためにその命令を命令待ち行列に置く。すべての必要な制御情報が受け取られる時、(メモリ・アクセス命令が、FXU49からのデータ位置合わせ及びエレメント・カウントのために下位アドレス・ビットを使用し)MXCU51において命令の実行が始まる。エレメント・カウントはマトリクス・カウント・レジスタ(MCR)53に保持され、ベクトル・メモリ・アクセス命令によってロード又は記憶されるべきエレメントの数を指定する。そこから、MXCU51は適当なコマンド(元の命令のフォーマット化バージョン)及び処理エレメント・マスク(命令を実行する場合に、どのMXPEが参加すべきかを識別する)を設定し、MXPEコマンド・バス70を介してマトリクス処理エレメントのアレイ(MXPE1-MXPEK)にそのコマンド(又は、一連のコマンド)をディスパッチして実行させる。

【0034】ベクトル・ロード命令に対しては、MXCU51は、L2キャッシュ35からMXUデータ・バス39を介して当該データを受け取り、そのデータを適当な目標MXPEデータ・バスへ経路指定してその目標MXPEに送る。ベクトル・ストア命令に対しては、MXCU51は、ソースMXPEからそれぞれのMXPEデータ・バスを介して当該記憶データを受け取り、その記憶データを適当に位置合わせするように経路指定し、そしてMXUデータ・バス39を介してL2キャッシュ35にそのデータを送る。ベクトル演算命令に対しては、MXCU51は、MXPEからMXPE状態バス72を介して結果状態(表示を除く)を受け取る。この状態は、マトリクス状態及び制御レジスタ(MXSCR)52において捕捉され、マスク可能例外条件が存在する場合に、MXU状態バス42を介してBRU57に割り込み要求の形で送られる。

【0035】MXPEはすべて、MXPEコマンド・バス70を介してコマンドを受け取る。MXCUは、各コマンドに従ってマスクを施し、どのMXPEが命令の実行に参加すべきかを識別する。MXPEのサブセットは、MXPEの数よりも少ないエレメント・カウントのような多数のファクタ、位置合わせされてないメモリ・アクセスによって選択可能である。即ち、プログラムは、MXSCU52の開始及び終了範囲フィールドにおいてサブセットを明瞭に指定する。マスク・オフされたMXPEはその命令を廃棄する。マスク・オフされてないMXPEは実行のための命令を受け取り、待ち行列に入れる。ベクトル・ロード命令に対しては、選択されたMXPEに対するロード・データが、ロード・コマンドによって、それぞれのMXPEデータ・バスを介して受け取られる。そこで、選択されたMXPEはロード・データを目標レジスタにロードする。ベクトル・ストア命令に対しては、選択された各MXPEはソース・レジスタをアクセスし、そのMXPEデータ・バスを介して

そのデータを与える。ベクトル演算命令に対しては、選択された各MXPEはそのリクエストされた演算機能をSIMD態様で遂行し、ソース・オペランドをマトリクス・レジスタ(MXR)から取得し、そしてその結果を目標MXRの中に置く。演算オペレーションに対する状態は、選択されたMXPEのマトリクス条件レジスタの各々の中に置き、そしてその生じた例外事項の概略がMXPE状態バス72を介してMXCU51に送られる。

【0036】図5は、MXCU51の編成の更に詳細な図を示す。MXCU51は、3つの主要なコンポーネント、即ち、命令制御装置(ICU)570、ロード・データ装置(LDU)560、及びストア・データ装置(SDU)550より成る。ICU570は、命令待ち行列装置571、コマンド発生装置572、及びコマンド・ディスパッチ装置573より成る。命令待ち行列装置571は、命令に対するすべての前提制御情報を受け取ってしまうような時間まで、CPU45からディスパッチされるMXU命令をバッファするために使用される。そのような前提データの例は、如何に多くのメモリ・アクセスがCPU45によって設定されようとしているかをMXU50に表示するロード命令及びストア命令のためのエレメント・カウントである。一旦、すべての必要な前提データが収集されてしまうと、その命令はコマンド発生装置572に進む。コマンド発生装置572は当該命令及び関連の制御情報を評価し、そしてMXPEにディスパッチされるべきその指定された機能を適切に且つ正しく実行する適当なコマンドを設定する。そのような設定は、MXSCR52の開始及び終了範囲設定に基づくMXPEマスクの生成、又はメモリ・アクセス情報のアドレス位置合わせ及びエレメント・カウントを含む。何れの特別のレジスタ・アドレッシング・モードもこのステージにおいて実施可能である。コマンド・ディスパッチ装置573はMXU50の現在の状態を評価し、次のコマンドがMXPEにディスパッチ可能であるかどうかを決定する。レジスタ従属性が可能となる場合の順序外ディスパッチのようなパフォーマンス強化技法がこのステージにおいて使用可能である。

【0037】LDU560は、ロード命令の実行準備ができる前に到達したMXUロード・データをデータ・バッファ504に与える。ここでは、ロード・データの順序外復帰をサポートするようなパフォーマンス強化技法がサポート可能である。そこで、LDU560は、任意のアービトラリMXPEデータ・バス・ポートへのロード・データの経路指定をサポートすることを意図したマルチプレクサのネットワーク、即ち、ルータ508を提供する。SDU550は、任意のアービトラリMXPEデータ・バス・ポートからのストア・データの経路指定をサポートすることを意図したマルチプレクサのネットワーク、即ち、ルータ534を提供する。MXPEデータ・バス74、76、78、及び80を停止することな

くMXUデータ・バス39のビジー期間を考慮するためにストア・データ・バッファ502が設けられる。

【0038】本発明によれば、MXU50は現在得られるベクトル・プロセッサに比べていくつかの明瞭な利点を与える。この利点を更に容易に理解するために、図2は現在得られるベクトル処理装置の概略図を示し、図3は本発明によるマトリクス処理装置の概略図を示す。図2及び図3の比較及び以下の説明は、それらの相違及び本発明の利点を指摘することになる。

【0039】図2は、計算機システム、即ち、メイン・メモリ及び中央処理装置202からデータ及び制御線204を介してデータ及び命令を受け取るベクトル処理装置200の概略図を示す。ベクトル処理装置200は、少なくとも1つの演算装置(AU)206を含んでいる。その演算装置206は、加算、乗算、及び論理的オペレーション等のようなオペレーションを遂行できる。ベクトル処理装置200は複数のベクトル・レジスタR1乃至RNを有する。但し、Nは1よりも大きい。従って、そのシステムはN個のベクトル・レジスタを有する。これらレジスタのうちの4つが、図2において、レジスタ208、210、212、及び214として示される。各レジスタR1乃至RNはワードW1乃至WKを含んでいる。但し、Kは1よりも大きく、各ワードは1ビットよりも大きいデータを有する。従って、各レジスタはK個のワードを含んでいる。Nは、典型的には、8乃至32個のレジスタ数である。データ又は命令はデータ線204、216、218、220及び222によって計算機システム202から転送される。各ワードは、各レジスタが長さKのベクトルを形成するようにK個のワードのデータで満たされるまで、ワード・ロケーションW1乃至WKに逐次に転送される。

【0040】例えば、レジスタ208をレジスタ210に加算しそしてその値をレジスタ212に置くようなコマンドがAU206によって実行される。各対応するレジスタ・ロケーションのワードが逐次に加算され、そしてその結果が図3における対応したワード・ロケーションに置かれる。例えば、レジスタR1のワード・ロケーションW1、即ち、ワード・ロケーションW1R1、におけるワードがデータ・バス224及び226を介してAU206に転送される。レジスタR2のワード・ロケーションW1、即ち、ワード・ロケーションW1R2、におけるワードは、データ・バス228及び230を介してAU206に転送される。W1R1+W1R2のオペレーションがAU206において遂行され、その結果は、データ線231及び232を介してレジスタR3のワード・ロケーションW1、即ち、ワード・ロケーションW1R3、に転送される。次のオペレーションは、W2R1におけるワードをAU206に転送すること及びW2R2におけるワードをAU206に転送することであり、AU206はW2R1+W2R2のオペレーショ

ンを遂行してその結果をワード・ロケーションW2R3に入れる。ワード・ロケーションW1乃至ワード・ロケーションWKまで、各レジスタR1、R2におけるワード・ロケーションから逐次転送が行われる。従って、メイン・メモリ及び中央処理装置202は、2つ以上の転送オペレーション及び2つ以上の演算オペレーションが一時に行われるように、2つ以上の演算装置AUを持つものであってもよい。

【0041】典型的には、一般に使用されるマトリクス・ベクトル処理装置は、精々4つの演算装置AUを持つものであり、それは、一時に4つのオペレーションを行うことができることを意味する。1つのベクトル・オペレーションは、レジスタR1におけるような1つのベクトルとレジスタR2におけるベクトルとの、加算、乗算又は論理的比較のような演算結合、及びR3のような他のベクトルへのその結果の転送である。そのようなベクトル命令又はオペレーションに対する機械サイクルの数は、そのような一般に得られるベクトル・プロセッサに対する5乃至10機械サイクルとベクトル長Kを演算装置AUの数により除したものと和である、1つのベクトル命令に対する始動サイクルに等しい。その演算装置は、この分野では、演算パイプとも呼ばれる。そのようなベクトル・プロセッサにおけるベクトル・オペレーションのための最小のサイクル・タイムは、パイプの数がベクトル長に等しい時、即ち、W1乃至WKのような各ワード・ロケーションがそれと関連付けられたパイプ又は演算装置を持つ時、である。そのような場合、1ベクトル命令当たりのサイクル・タイムは、1つのベクトル・オペレーションに対して、5乃至10機械サイクル+1である始動時間に等しい。1ベクトル命令当たりの最大のサイクル数は、パイプの数が1に等しい時に生じ、その場合、1ベクトル命令当たりのサイクル数は、5乃至10機械サイクル+ベクトル長Kである始動サイクルに等しい。

【0042】図2に概略的に示される現在利用可能なベクトル処理装置は、計算機システム202及びベクトル処理装置200の間のデータ帯域幅が不十分で、K個の演算装置(AU)206の能力を利用するに十分な程速くデータ又は命令を供給できないため、K個の演算装置206を持つことができない。

【0043】拡張帯域幅のメモリは、ベクトル・オペレーション相互間で5乃至10サイクルの最少時間を得るためには、K個の演算装置206を利用する必要がある。Qが1つの演算装置206に対する帯域幅である場合、K個の演算装置206に対する帯域幅はK*Qでなければならない。そのような拡張帯域幅のメモリは、大きく増加した複雑性と、それによるコストとを必要とする。反対に、本発明によるプロセッサは、拡張帯域幅を必要とせずにベクトル・オペレーション相互間で1サイクル・タイムを得るものであり、Qの帯域幅でもってこ

れを得ることができる。

【0044】1つのベクトルの負の逆元又は絶対値、2つのベクトルの加算又は乗算、又はこれらの演算オペレーションの組合せのようなベクトル・オペレーションが、少なくとも1つ(代表的には、2つ又は3つ)のベクトルにおいて遂行される。ベクトル・オペレーションは、2つのベクトルを乗算しそして第3のベクトルを加算するオペレーション(1つのベクトル・オペレーションで行われる)のような3つ以上のベクトルに関するものであってもよい。

【0045】本発明によるベクトル処理装置、又はマトリクス処理装置300が図3に300として概略的に示される。処理装置300は、計算機システム302からデータ及び制御線304を介してデータ及び命令を受け取る。N個のベクトルに対応したN個のレジスタR1乃至RNがあり、Kのベクトル長に対してこれらレジスタの各々におけるK個のワードW1乃至WKがある。レジスタR1乃至RNの各々における第1ワード・ロケーションW1はすべて、303として示された処理エレメント1(PE1)に含まれている。処理エレメント303は演算装置1(AU1)305を含む。同様に、レジスタR1乃至RNの各々における第2ワードW2は、306として指定された処理エレメントPE2に含まれる。処理エレメント306は、308として示された演算装置AU2を含む。同様に、それらベクトルの各々における各ワードエレメントWI(但し、IはKまでである)は、それ自身の演算装置を含む処理エレメント内に含まれる。各ワード・エレメントWIは、Mビット・ワードに対するMビットのデータを記憶するためのM個のロケーションを有するレジスタに含まれる。各々がK個のワードを有するN個のベクトルは、K*Nワード・マトリクスを形成する。従って、ここでは、図3のプロセッサはマトリクス・プロセッサと呼ばれる。

【0046】図1に戻ると、マトリクス装置50は、マトリクス制御装置MXCU51及びK個の処理エレメントMXPE1乃至MXPEKを含む。これらマトリクス処理エレメントのうちの6つは数字60、61、62、63、64、及び65によって識別される。マトリクス制御装置MXCU51のエレメント52は、コマンド・バス70及び複数個のデータ制御バス68を介してマトリクス・エレメントMXPE1乃至MXPEKの各々に接続される。コマンド・バス70は、マトリクス制御装置52から処理エレメントMXPE1乃至MXPEKの各々にコマンドを搬送する。図1では、データ制御バス68は、エレメント74、76、78、及び80として示される。図1に示されるように、データ制御バス68のうちのいくつかは、2つ以上のマトリクス処理エレメントMXPEの間で共用可能である。例えば、データ制御バス80は、データ制御線81及び82によって示されるように、それぞれ、マトリクス処理エレメントMX

PE1及びMXPE5の間で共用される。ベクトル・レジスタ402は、図3における単一の処理エレメント303のレジスタ、例えば、ワード・ロケーションW1R1乃至W1RNに対応する。

【0047】図5は、図1のマトリクス制御装置(MXC)51の概略図を示す。命令は、MXU50から命令バス41を介して制御装置570により受け取られる。それは、命令のタイプに基づいて、データをストア・データ・バッファ502に記憶すべきか、又はロード・データ・バッファ504に記憶すべきかというそれを扱う方法を決定する。データは、CPUデータ・バス39を介してMXU50に及びMXU50から転送される。CPUデータ・バス39を介して到着するデータはマルチプレクサ501に入る。そのマルチプレクサは、CPU45から情報を受け取る時、バス506を介してロード・データ・バッファ504にそのデータを経路指定する。ロード・データ・バッファ504におけるデータは、スイッチ、即ち、ルータ508によって、図1のデータ制御バス68に対応したデータ・バス510、512、514、及び516に分けられる。各データ・バス510、512、514、及び516は、それぞれ、マルチプレクサ518、520、522、及び524に通じ、そしてそれらマルチプレクサから、データがデータ・バス74、76、78、及び80に転送される。データ・バス74、76、78、及び80におけるそのデータは、マトリクス処理エレメントMXPE1乃至MXPEKに転送される。

【0048】一方、データがマトリクス処理エレメントMXPE1乃至MXPEKから転送される時、そのデータはデータ・バス74、76、78、及び80を介して、それぞれ、マルチプレクサ518、520、522、及び524に転送され、そしてそこから、それぞれ、データ線526、528、530、及び532を介してスイッチ、即ち、ルータ534に送られる。データは、そのスイッチ534からストア・データ・バッファ502に転送され、更に、データ線535、マルチプレクサ501、及びデータ・バス39を介して図1のCPU45に転送される。マルチプレクサ501は、1つの双方向入力、即ち、データ・バス39と、2つの単方向データ線535及び506を有する。

【0049】図4は、図1のMXU50におけるMXPEの1つの概略図を更に詳細に示す。データ・バス401は、図1のデータ・バス74、76、78、又は80のうちの任意のものでよい。NICAがそのシステムの残りのもの(メモリ)と対話する方法を以下で説明する。本発明によるマトリクス装置の1つの主要な目的は、メモリ帯域幅(BW)要件を減少させることである。これは、NICAレジスタにロードされたデータの有意な再使用によって達成可能である。従って、NICAとメモリとの間の帯域幅(BW)に関する設計ポイ

ントは、1サイクル当たりのベクトル・ロード/ストア(Kオペレーション)よりもかなり少なくなければならない。ベクトル・ロード/ストアは、多数のサイクルにおいて1回の割合でしか起こらないものと予測される。

【0050】この仮定の場合、NICAとシステムとの間のデータ転送を2つの要素に分割することができる。第1の要素は、システムとNICAロード・ストア/バッファ(LSB)400との間の非同期転送である。第2の要素は、LSB400とNICAベクトル・レジスタ402との間の同期(SIMD)転送である。ちょうどベクトル・レジスタのように、LSB400も、MXU50におけるすべてのMXPEに分布している。LSB400における1つのMXPE当たりの与えられたロケーションの数がもう1つの設計パラメータである。ベクトル・ロードに対して、システムは、メモリにおけるオペランドのアドレスを計算し、それらをフェッチし、そして宛先MXPEのインデックスをもって到着データをタグし、それらをバス上に(高い帯域幅に対しては、各バスがMXPEのサブセットにだけ接続されて成る複数のバスが使用可能である)置く。宛先MXPEは到着データをラッチし、それをLSB400の指定されたロケーションに置く。すべてのMXPEがそれらのデータを受け取った時、MXPE制御装置404は、LSB400からその指定されたベクトル・レジスタへのデータのSIMD転送を開始する。ベクトル・ストアは同様に、しかし、逆に作用する。データ再使用のために、ベクトル・ロード/ストアは減多にないことであり、NICAはレジスタにおけるデータに関してレジスタ・ツー・レジスタ(RR)ベクトル演算を行い続けるものと考えられる。

【0051】次のものは拡張の例示リストである。これらの思想のいくつか又はすべてがNICAにおいて利用可能である。即ち、

○「スカラ・レジスタ」のセットは、別個のスカラ・レジスタ・バスを介してすべてのMXPEに対して利用可能にされる。バスはスカラ・レジスタのプールから送られ、すべてのMXPEは各MXPE上の別個の「スカラ・ポート」を介してそれに接続される。このバスは、すべてのPEの演算装置にスカラ定数を与えるために使用可能である。このバスは、ベクトル・レジスタにおける定数の速いローディングのためにも使用可能である。

○多数のPEが単一のチップ/モジュール上に製作可能である。これは、すべてのバスに対するI/OピンがPE間で共用可能であるという利点を有する。

○M個の1ビット・ベクトル・マスク又は条件レジスタのセットを設けることが可能である。これらは、再び、各PEがM個の1ビット・マスク又は条件レジスタのセットを有する場合、すべてのPEに分配される。これらのレジスタは、比較によって、又は演算オペレーションの条件コードによってセット可能である。それらは、P

Eとそのシステムとの間のデータ移動を実施するために並びにPE内での条件付き演算及びデータ移動を実施するために使用可能である。これらのビット自体は、システムとNICAとの間で双方向に転送可能である。ベクトル条件レジスタとシステムとの間のビット転送のために、別個のKビット幅の並列バスが使用可能である。各PEは個の並列バスのうちの1ビットを得る。これらのマスクがNICAとシステムとの間のデータ転送を制御するために使用される場合、制御装置にとってこのバスが利用可能であることは、その転送のための制御データの速い先回り処理を助けるであろう。マスク・ベクトルは1ビットのブール演算によっても操作可能である。

○バック／アンバック・オペレーションを行うと（例えば、2つの短精度オペランドを1つの長精度オペランドとしてバックする）、短精度データはバス上を更に効率的に転送可能である。これは、利用可能なメモリ及びバス帯域幅の効率的な利用を助ける。

○SIMDベクトル命令に参加するMXPEの範囲は、アクティブなMXPEの開始及び終了インデックスを指定する「ベクトル範囲設定」命令によって制御可能である。この範囲内であっても、対応するMXPEにおける指定されたマスク・ビット（M個のマスク・ビットの1つ）が「1」である場合しか目標は更新されない。

【0052】NICAがNIC問題を解決する場合に適可能であるためには、2つの主要な必要条件、即ち、ベクトル化及びデータ再使用、がある。最も伝統的なベクトル・オペレーションは、NICAにおいてSIMD態様で実施可能である。しかし、いくつかの例外がある。1つのベクトルにおける相異なるエレメントの間の相互作用を必要とするベクトル・オペレーションをSIMD態様で行うことはできない。そのような命令の重要な例はベクトル累算（乗算累算は累算と同じである）である。累算では、ベクトルのサム・リダクション（sum reduction）が行われる。演算パイプラインのため、ベクトル・マシン上でも、これはあまり効率的ではない。非常に長いベクトルを累算する場合、そのベクトルの8つ毎のエレメントがアキュムレータの1つにおいて累算される。オペレーションのこのフェーズは十分なベクトル速度で完了する。これに続いて、「部分和の合計」オペレーションが生じる。それはスカラ・モードで行われる。累算（又は、乗算累算）は、部分和の数がKに等しくされた場合、NICA上でも実施可能である。次に、非常に長いベクトルに対しては、殆どの計算が高パフォーマンスのSIMD態様で行われるであろう。最後の「部分和の合計」位相だけがスカラ計算で行われるであろう。乗算累算オペレーションを回避するために、しばしば、殆どのアルゴリズムが再公式化可能であり、そしてその代わりに、乗算加算オペレーションが使用可能である。

【0053】SIMD計算を施すことが可能でないもう

1つのオペレーションは、ベクトルの最小及び最大を見つけることである。ここで、再び、問題サイズをKまで縮小するためにSIMDベクトル比較が使用可能である。従って、その作業の殆どが高パフォーマンスSIMDモードで遂行可能である。

【0054】前述のように、ベクトル化はNICAを使用する場合の前提である。殆どのNICAアプリケーションが内部ループ・レベル又は外部ループ・レベルでベクトル化する。NICAが非常に効果的であるためには、ベクトル・レジスタに与えられたデータの有意な再使用が必要である。再使用は、浮動小数点演算対データ比（FDR）に関連する。当初の問題を或レベルでブロック化することによって、多くの問題に対して、FDRは大いに改良可能である。1つの有用な技法は外部ループ・ベクトル化である。その場合、内部ループ又はこれらのサブブロックがNICAレジスタに適合する。そこで、外部ループがKのブロック・サイズでもってブロック化され、そしてすべての内部ループ計算が1つのPEにおいて生じる時、外部ループの各インデックスが別個のPEにおいて実行される。これは、外部ループ・ベクトル化／並列化の非常に一般的な形式である。多くの並列化コンパイラが、この技法を使用して多くの並列プロセッサに跨って計算を分配する。ベクトル化し得ない多くの問題が、しばしば、並列化可能である。例えば、地震の計算では、複雑な三重対角等式の多重システムを解く場合に、大量の計算が行われる。これらは、パフォーマンスにおける非常に大きな利得をもって、NICAにおいて実施可能である。大量の地震の計算はトレース・ベース化され、しばしば、それらトレースに跨って並列化可能である。

【0055】FORTRAN-90と呼ばれるFORTRANの拡張が定義されている。それは、アレイ演算を行う機能を提供する。FORTRAN-90で書かれたプログラムはNICAにとって容易にコンパイル可能である。

【0056】LAPACKは、線形代数計算を行うための公知のソフトウェアである。それは、LINPACK及びEISPACKパッケージに取って代わるものであり、それらを大いに拡張するものである。それは、殆どの現在利用可能なコンピュータにおける非常に高いパフォーマンスのために書かれている。その高いパフォーマンスのために、線形代数サブルーチン及びNAGのような他の商業計算ソフトウェア・パッケージの殆どのユーザが、LAPACKに移行している。LAPACKプロジェクトの背後にある重要な思想は、殆どの計算がBLAS-3で行われない場合、高パフォーマンスを得ることはできないということである。BLAS-3は、 N^2 のオーダーのデータ・ポイントで、 N^3 のオーダーの計算を行う線形代数カーネルのセットである。LAPACKコードは移植可能であり、コンピュータ製造者によって

提供された調整された（特殊機械用に）BLAS-3カーネルを利用する。この思想及びソフトウェア・パッケージは、共用メモリ並列プロセッサ及びMIMD機械まで更に拡張される。BLAS-3ルーチンの調整されたセットがNICA用に開発されることが期待される。そこで、LAPACKサブルーチン・コールを持ったユーザ・コードは、ユーザ部分において如何なる努力も払うことなく、NICAからパフォーマンス・ブーストを自動的に得るであろう。

【0057】一方、BLAS-3ルーチンは、DGEMM（更新を伴うマトリクス・マトリクス乗算）においてその計算の殆どを行う。DGEMMにおいて行われない計算は、それが三角マトリクスにおいて行われることを除けば同じである。以下では、NICAにおいて非常に効率的にDGEMMを実施できる方法を示す。

【0058】DGEMMは次の計算、即ち、

$$C \leftarrow C + A * B$$
 を行う。

【0059】サイズが $K * P$ である C というブロックが一時に計算されるように、 C に関する何らかの外部レベル・ブロッキングがあるものと仮定する。但し、 P は N よりも小さく、NICAにおいて利用可能なベクトル・レジスタの数である。その場合、 C マトリクスのこのブロックはNICAレジスタに適合する。 A はサイズ $K * L$ のものであると仮定する。但し、 L はその問題の中間の大きさである。そこで、上記の計算は、 C マトリクスに関するランク L 更新を行うことになる。これは、 L 個のステップで行うことができる。各ランク L 更新のためには、 A マトリクス（ K 個のエLEMENT）の1つの列と B マトリクス（ P 個のエLEMENT）の1つの行が必要である。これら $K + P$ 個のエLEMENTがNICAに与えられ、 $2 * P * K$ 浮動小数点演算を生成するために使用される。この問題に対する浮動小数点演算対データの比は、 $2 / (1/K + 1/P)$ である。 K 及び P の両方とも大きい場合、これは非常に計算集中的な問題となる。NICAとシステムとの間のメモリ帯域幅は、 $2 * K * P$ 浮動小数点演算を行いながら（ $K + P$ ）データ転送をサポートするに十分な大きさでなければならない。 $2 * K$ 浮動小数点演算が1サイクルで行い得るものであると仮定すると、1サイクル当たり少なくとも $(1 + K/P)$ 個のエLEMENTのデータ転送速度を必要とする。これは、 C マトリクスの初期ローディング及び最終ストアリングを考慮していない。 L が大きい場合、この余分なオーバーヘッドは小さくなる。

【0060】1次元及び多次元FET、多重シーケンスの回旋／相関、地震の処理における3次元移行等のような多くの他の重要な計算カーネルが知られている。これらアプリケーションはすべて、NICAから大きなパフォーマンス利得を得ることができる。一般的な稠密、正定値の対称、複体対称、帯域（VSSのオーダの、又は

更に長い帯域幅を持つ）、スカイライン等のような種々な種類の式を解くことは、すべて、ピークに近いパフォーマンスでもってNICA上で実施可能である。汎用稠密システムの等式（サイズ1000又はそれ以上）を解くLINPACK TPPベンチマークも、ピーク・パフォーマンスに近くなるであろう。これは非常に重要な超高速計算ベンチマークであり、スーパーコンピュータは、しばしば、この基準で比較される。

【0061】大規模な科学計算におけるもう1つの傾向は、非常に大きな疎システムの等式に向いている。これらの問題を解くために、しばしば、直接方法が使用される。直接疎方法は疎BLAS-3（現在の標準的活動はこれらBLASを定義する用に進んでいる）によって公式化可能である。一方、疎BLAS-3は、NICAに対してブロック化可能である。疎マトリクスは、抽出されたブロックが全く稠密に見えるような方法で行及び列のセットがそのマトリクスから抽出されるように再配列可能である。疎マトリクス全体は、これらブロック及び他の構造体（対角線の帯域のような）のセットの線形合計として表される。これらは稠密ブロックとしてNICAベクトル・レジスタに転送され、何回も使用可能である。

【0062】高パフォーマンスのグラフィックスも高いFDRを持つ傾向があり、ベクトル化可能である。これらのアプリケーションもNICAから利益を得ることができる。

【0063】まとめとして、本発明の構成に関して以下の事項を開示する。

【0064】（1）プロセッサ・サイクル・タイムを有する計算機プロセッサにして、 K 個の処理エレメント（但し、 $K > 1$ ）にして、前記 K 個の処理エレメントの各々は N 個のレジスタのセット（但し、 $N > 1$ ）及び演算装置を含み、前記レジスタの各々の I 番目のエレメントはベクトル長 K の I 番目のベクトルのための記憶ロケーションを提供し（但し、 $1 \leq I \leq N$ ）、それによって、 N 個のベクトルを提供するものと、前記 N 個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記 N 個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記 N 個のベクトルの少なくとも1つにおける K 個のエLEMENTすべてに対して実質的に同時に前記 K 個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機プロセッサ。

（2）前記 N 個のレジスタのセットにおける各レジスタは M 個のビット記憶ロケーション（但し、 $M > 0$ ）を有することを特徴とする上記（1）に記載の計算機プロセッサ。

(3) 前記ベクトル・オペレーションを遂行するための手段である第1演算装置及び前記他のベクトル・オペレーションを遂行するための手段である第2演算装置は同じ演算装置であることを特徴とする上記(1)に記載の計算機プロセッサ。

(4) マトリクス制御装置を含むことを特徴とする上記(1)に記載の計算機プロセッサ。

(5) 前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスを含むことを特徴とする上記(4)に記載の計算機プロセッサ。

(6) 前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスを含むことを特徴とする上記(4)又は(5)に記載の計算機プロセッサ。

(7) 前記少なくとも1つのデータ・バスは前記K個の処理エレメントのうちの2個以上の処理エレメントによって共用されることを特徴とする上記(5)に記載の計算機プロセッサ。

(8) サイクル・タイムを有する計算機プロセッサにして、マトリクス制御装置と、前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段と、を含む計算機プロセッサ。

(9) 前記手段はベクトル演算オペレーションを遂行することを特徴とする上記(8)に記載の計算機プロセッサ。

(10) 前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスと、前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスと、を含むことを特徴とする上記(8)に記載の計算機プロセッサ。

(11) サイクル・タイムを有する計算機システムにして、メイン・メモリと、キャッシュ・メモリと、記憶制御装置と、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、K個の処

理エレメント(但し、 $K > 1$)にして、前記K個の処理エレメントの各々はN個のレジスタのセット(但し、 $N > 1$)及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し(但し、 $1 \leq I \leq N$)、それによって、N個のベクトルを提供するものと、前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機システム。

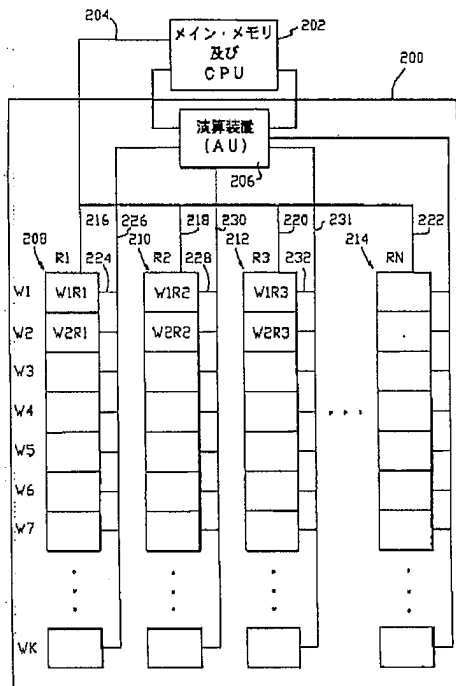
(12) サイクル・タイムを有する計算機システムにして、メイン・メモリと、キャッシュ・メモリと、記憶制御装置と、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、を含み、前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、前記マトリクス処理装置は、K個の処理エレメント(但し、 $K > 1$)にして、前記K個の処理エレメントの各々はN個のレジスタのセット(但し、 $N > 1$)及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し(但し、 $1 \leq I \leq N$)、それによって、N個のベクトルを提供するものと、前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含むことを、特徴とする計算機システム。

(13) マトリクス制御装置と、前記マトリクス制御装

置を前記K個の処理エレメントに接続するためのコマンド・バスと、前記マトリクス制御装置を前記K個の処理エレメントに接続するための少なくとも1つのデータ・バスと、を含むことを特徴とする上記(12)に記載の計算機システム。

(14) サイクル・タイムを有する計算機システムにして、メイン・メモリと、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、前記メイン・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、前記中央処理装置は第1データ・バスによって前記メイン・メモリに接続する手段と、前記マトリクス制御装置は第2データ・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むこと、K個の処理エレメント(但し、 $K > 1$)にして、前記K個の処理エレメントの各々はN個のレジスタのセット(但し、 $N > 1$)及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し(但し、 $1 \leq I \leq N$)、それによって、N個のベクトルを提供する

【図2】



ものと、前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機システム。

【0065】

【発明の効果】本発明により、計算機システムのメモリ帯域幅を大きく増大させることなく、制限された数の命令をより速く実行することが可能となる。

【図面の簡単な説明】

【図1】本発明の実施例に従って構成されたデータ処理システムのブロック図を示す。

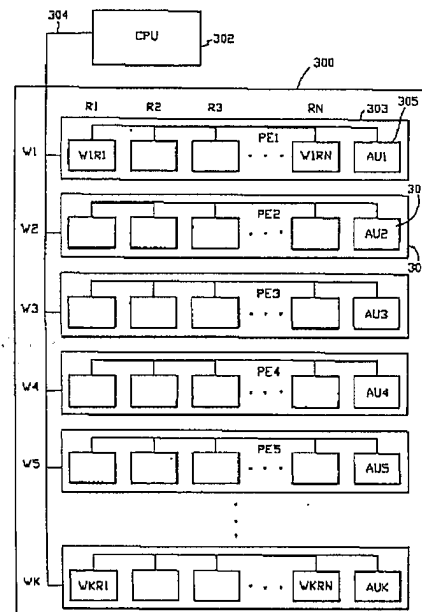
【図2】一般に利用可能なベクトル・プロセッサの概略図を示す。

【図3】本発明によるベクトル・プロセッサの概略図である。

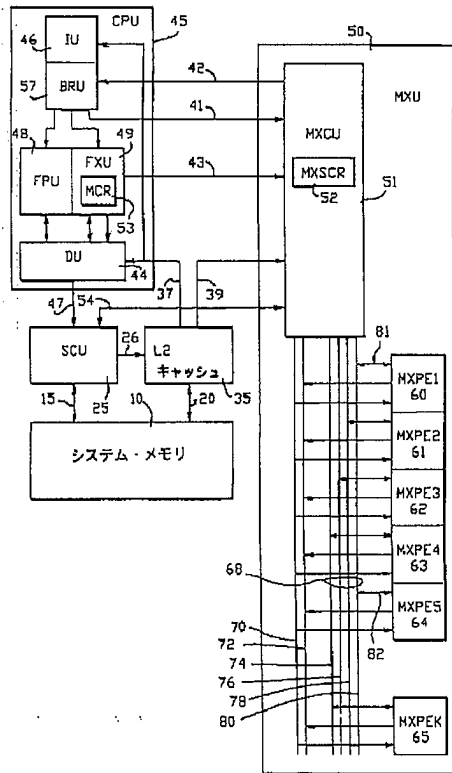
【図4】図3のベクトル・プロセッサの処理エレメントMXPEの1つの詳細なブロック図である。

【図5】図3のベクトル・プロセッサのマトリクス制御装置MXUの詳細なブロック図である。

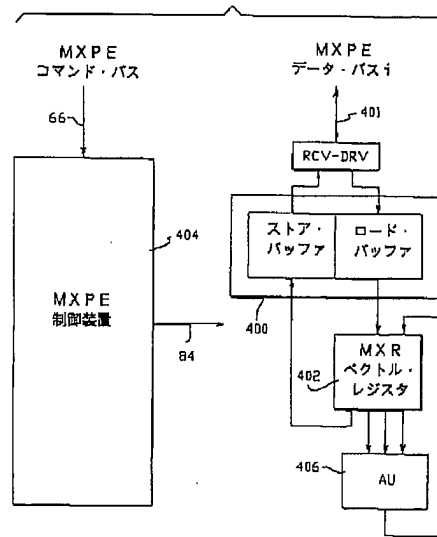
【図3】



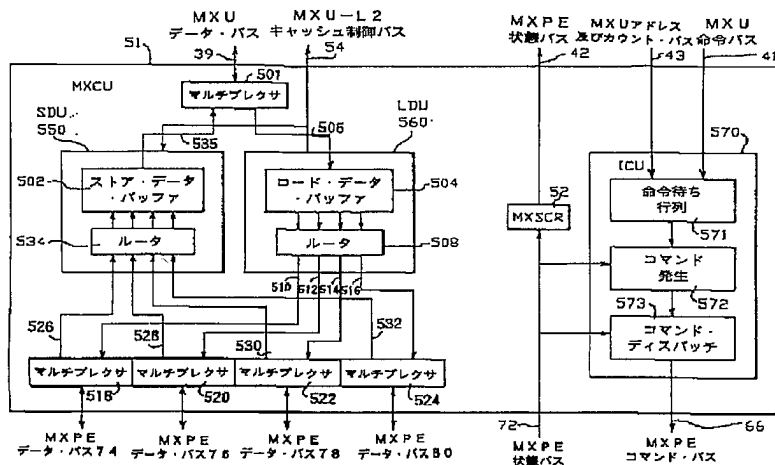
【図1】



【図4】



【図5】



フロントページの続き

(72)発明者 ランダル・ディーン・グローブス
アメリカ合衆国テキサス州、オースティ
ン、ミドルピーパー・ドライブ 9002
(72)発明者 フレッド・ゲールング・ガスタブソン
アメリカ合衆国ニューヨーク州、ブリアク
リフ・マノア、サウス・ステイト・ロード
70

(72)発明者 マーク・アラン・ジョンソン
アメリカ合衆国テキサス州、オースティ
ン、タレイラン・ドライブ 10105
(72)発明者 ブレット・オルッソン
アメリカ合衆国テキサス州、ラウンド・ロ
ック、シンコタニュー・ウェイ 1800